

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

Control de autenticación y
acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

AdictosAlTrabajo

Terrakas 1x05
¡¡Ya está en la web!! :-)
terrakas.com



autentia
Soporte a desarrollo informático
Hosting patrocinado por
enredados

Entra en Adictos a través de  

E-mail

Contraseña

Entrar Deseo registrarme
Olvidé mi contraseña

[Inicio](#) [Quiénes somos](#) [Formación](#) [Comparador de salarios](#) [Nuestros libros](#) [Más](#) 

» Estás en: Inicio » Tutoriales » jBPM5 Console Server and Human Task Server: instalación y configuración



Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en [Autentia](#): Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE



[Ver todos los tutoriales del autor](#)

Catálogo de servicios Autentia



Fecha de publicación del tutorial: 2013-03-11

Tutorial visitado 1 veces [Descargar en PDF](#)

jBPM5 Console Server and Human Task Server: instalación y configuración

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Instalación.
- 4. Integración con LDAP.
- 5. Referencias.
- 6. Conclusiones.

1. Introducción

Es este tutorial vamos a revisar cómo llevar a cabo una instalación personalizada de los componentes que dan soporte al core de jBPM5:

- jBPM Console Server: es la pieza principal que da soporte a la gestión de procesos, proporciona un api REST que nos permite además realizar todas las operaciones relacionadas con la gestión de tareas, de usuarios y la renderización de formularios. Podríamos decir que es el API de gestión de procesos.
- jBPM Human Task Server: es una pieza nueva que hace de repositorio de tareas y que permite además realizar todas las operaciones necesarias para la gestión de usuarios y formularios. En realidad el API REST que nos proporciona el jBPM Console Server no es más que una fachada de estos servicios a los que accede mediante JMS.
- JBPM Console: es una aplicación web que nos permite acceder a la lista de tareas pendientes por usuario o grupo, arrancar un proceso y monitorizarlo. Sería la pieza que podríamos incrustar en nuestras aplicaciones o tomar de referencia para hacer el mismo uso del API REST del Console Server para desarrollar un bandeja de tareas propia, con la tecnología que estimemos oportuna, puesto que no sería más que comunicarnos vía REST.

En este tutorial vamos a realizar una instalación de estos 3 componentes de forma controlada, fuera de la [demo de instalación de jBPM5](#), con el soporte de Maven, sobre un Jboss AS 7. Se podría realizar también la instalación sobre un [Apache Tomcat](#) pero ya tenemos sobre Apache Tomcat desplegados [Guvnor](#), el Designer y aún nos queda desplegar el generador de formularios con lo que vamos a diversificar la carga. Vamos a dejar el núcleo de los servicios, aquellos que necesitan persistencia a base de datos y comunicación por colas, en Jboss Server.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 15' (2.4 GHz Intel Core i7, 8GB DDR3 SDRAM).
- Sistema Operativo: Mac OS X Lion 10.7.4
- jBPM 5.4.2.Final
- JBoss AS 7.1.1

3. Instalación.

Con el soporte de Maven vamos a añadir vamos a añadir 3 proyectos más a nuestro proyecto `tnt-jbpm`, que ya vimos en el [tutorial de Guvnor](#):



Síguenos a través de:



Últimas Noticias

» Vendedor: Soy inseguro, filtra o elige por mí: si quieres que te compre.

» [Comentando el libro: El arte de pensar, de Rolf Dobelli](#)

» [Ya está a la venta mi segundo libro: Planifica tu éxito, de aprendiz a empresario](#)

» Ya esta disponible en eBook mi primer libro: Informática Profesional

» [Comentando el libro: La inteligencia reformada, las inteligencias múltiples en el siglo XXI de Howard Gardner](#)

[Histórico de noticias](#)

Últimos Tutoriales

» [Spring Security: haciendo uso de un servidor LDAP embebido.](#)

» [Introducción a Guvnor](#)

» [Gestión de expedientes en el ámbito de las](#)

```

1 <modules>
2   <module>tnt-designer</module>
3   <module>tnt-guvnor</module>
4   <module>tnt-jbpm-gwt-console-server</module>
5   <module>tnt-jbpm-human-task</module>
6   <module>tnt-jbpm-gwt-console</module>
7 </modules>

```

3.1. tnt-jbpm-gwt-console-server.

El proyecto tnt-jbpm-gwt-console-server tendría el siguiente pom.xml:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5
6   <parent>
7     <groupId>com.autentia.bpm.repository</groupId>
8     <artifactId>tnt-jbpm</artifactId>
9     <version>0.0.1-SNAPSHOT</version>
10   </parent>
11
12   <groupId>com.autentia.bpm.repository</groupId>
13   <artifactId>tnt-jbpm-gwt-console-server</artifactId>
14   <version>0.0.1-SNAPSHOT</version>
15   <packaging>war</packaging>
16
17   <inceptionYear>2013</inceptionYear>
18
19   <build>
20     <finalName>gwt-console-server</finalName>
21     <plugins>
22       <plugin>
23         <groupId>org.apache.maven.plugins</groupId>
24         <artifactId>maven-compiler-plugin</artifactId>
25         <version>2.5.1</version>
26         <configuration>
27           <source>${java.version}</source>
28           <target>${java.version}</target>
29         </configuration>
30       </plugin>
31       <plugin>
32         <groupId>org.apache.maven.plugins</groupId>
33         <artifactId>maven-surefire-plugin</artifactId>
34         <version>2.4.2</version>
35         <configuration>
36           <skipTests>true</skipTests>
37         </configuration>
38       </plugin>
39       <plugin>
40         <groupId>org.apache.maven.plugins</groupId>
41         <artifactId>maven-war-plugin</artifactId>
42         <configuration>
43           <dependentWarExcludes>WEB-INF/lib/netty-*.jar, WEB-INF/lib/hibernate*
44           <archive>
45             <manifestEntries>
46               <dependencies>org.jboss.netty</dependencies>
47             </manifestEntries>
48           </archive>
49         </configuration>
50         <version>2.3</version>
51       </plugin>
52     </plugins>
53   </build>
54
55   <dependencies>
56     <dependency>
57       <groupId>org.jbpm</groupId>
58       <artifactId>jbpm-gwt-console-server</artifactId>
59       <version>5.4.0.Final</version>
60       <type>war</type>
61     </dependency>
62   </dependencies>
63
64 </project>

```

La única dependencia es con el proyecto maven que distribuye el API y dentro del plugin de generación del war tenemos que:

- emular el contenido del **assembly** que está preparado para la generación del war en un entorno JEE excluyendo aquellas librerías que ya existen en Jboss AS
- añadir al MANIFEST.MF la dependencia de la librería org.jboss.netty necesaria para la comunicación vía JMS.

En nuestro proyecto, vamos a añadir la configuración de la unidad de persistencia para que cree y almacene la información sobre los procesos en una base de datos MySQL. Así, dentro del directorio src/main/resources/META-INF, vamos a incluir un fichero persistence.xml con el siguiente contenido:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <persistence version="2.0"
3   xmlns="http://java.sun.com/xml/ns/persistence"
4   xmlns:orm="http://java.sun.com/xml/ns/persistence/orm"
5   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6   xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/xml/ns/
7
8   <persistence-unit name="org.jbpm.persistence.jpa" transaction-type="JTA">
9     <provider>org.hibernate.ejb.HibernatePersistence</provider>
10    <jta-data-source>java:jboss/datasources/jbpm5DS</jta-data-source>
11    <mapping-file>META-INF/JBPMorm.xml</mapping-file>

```

Administraciones Públicas (IV): buscando en las forjas una solución.

» Gestión de expedientes en el ámbito de las Administraciones Públicas (III): BPM y la gestión de procesos de negocio.

» jBPM5: usando nuestra propia base de datos.

Últimos Tutoriales del Autor

» Spring Security: haciendo uso de un servidor LDAP embebido.

» Introducción a Guvnor

» Gestión de expedientes en el ámbito de las Administraciones Públicas (IV): buscando en las forjas una solución.

» Gestión de expedientes en el ámbito de las Administraciones Públicas (III): BPM y la gestión de procesos de negocio.

» jBPM5: usando nuestra propia base de datos.

Últimas ofertas de empleo

2011-09-08

 Comercial - Ventas - MADRID.

2011-09-03

 Comercial - Ventas - VALENCIA.

2011-08-19

 Comercial - Compras - ALICANTE.

2011-07-12

 Otras Sin catalogar - MADRID.

2011-07-06

 Otras Sin catalogar - LUGO.



Jose Manuel Sánchez
sanchezsuaresj

sanchezsuaresj @aamormo
muchas gracias hombre :)
6 days ago · reply · retweet · favorite

sanchezsuaresj Spring Security: haciendo uso de un servidor LDAP embebido. - kcy.me/gdem
6 days ago · reply · retweet · favorite

adictosaltrabaj Introducción a Guvnor - kcy.me/g8hs como repositorio central de recursos dentro de nuestro ecosistema bpm #jbpm
9 days ago · reply · retweet · favorite

adictosaltrabaj Gestión de expedientes en el ámbito de las Administraciones Públicas (IV): buscando en las forjas una solución. - kcy.me/g0de



Join the conversation

```

12 <mapping-file>META-INF/ProcessInstanceInfo.hbm.xml</mapping-file>
13 <mapping-file>META-INF/ExtraIndexes.hbm.xml</mapping-file>
14
15 <class>org.jbpm.persistence.processinstance.ProcessInstanceInfo</class>
16 <class>org.drools.persistence.info.SessionInfo</class>
17 <class>org.drools.persistence.info.WorkItemInfo</class>
18 <class>org.jbpm.process.audit.ProcessInstanceLog</class>
19 <class>org.jbpm.process.audit.NodeInstanceLog</class>
20 <class>org.jbpm.process.audit.VariableInstanceLog</class>
21 <properties>
22 <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect"/>
23 <property name="hibernate.max_fetch_depth" value="3"/>
24 <property name="hibernate.hbm2ddl.auto" value="update" />
25 <property name="hibernate.show_sql" value="true" />
26 <property name="hibernate.transaction.manager_lookup_class" value="org.hibernate.t
27
28 <!-- BZ 841786: AS7/EAP 6/Hib 4 uses new (sequence) generators which seem to cause
29 <property name="hibernate.id.new_generator_mappings" value="false" />
30
31 </properties>
32 </persistence-unit>
33
34 </persistence>

```

Y, lo último, será crear un fichero `default.jbpm.console.properties` en el directorio `src/main/resources` con el siguiente contenido:

```

1 jbpm.console.server.host=localhost
2 jbpm.console.server.port=8081
3 jbpm.console.task.service.strategy=HornetQ
4 jbpm.console.task.service.host=127.0.0.1
5 jbpm.console.task.service.port=5153
6 jbpm.console.directory=
7 guvnor.protocol=http
8 guvnor.host=localhost:8080
9 guvnor.subdomain=drools-guvnor
10 guvnor.usr=admin
11 guvnor.pwd=admin
12 guvnor.packages=
13 guvnor.connect.timeout=10000
14 guvnor.read.timeout=10000
15 guvnor.snapshot.name=LATEST

```

En este fichero se configuran las comunicaciones del API hacia:

- la consola jBPM: la idea es desplegar Jboss por el puerto 8081, para no colisionar con el puerto de Apache Tomcat,
- el servidor de tareas, que va vía JMS,
- el repositorio de Guvnor

Salvo la modificación del puerto `jbpm.console.server.port`, el resto de propiedades son las de por defecto.

3.2. `tnt-jbpm-human-task`.

Igualmente, veamos el `pom.xml`:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XML
3 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/m
4 <modelVersion>4.0.0</modelVersion>
5
6 <parent>
7 <groupId>com.autentia.bpm.repository</groupId>
8 <artifactId>tnt-jbpm</artifactId>
9 <version>0.0.1-SNAPSHOT</version>
10 </parent>
11
12 <groupId>com.autentia.bpm.repository</groupId>
13 <artifactId>tnt-jbpm-human-task</artifactId>
14 <version>0.0.1-SNAPSHOT</version>
15 <packaging>war</packaging>
16
17 <inceptionYear>2013</inceptionYear>
18
19 <build>
20 <finalName>jbpm-human-task</finalName>
21 <plugins>
22 <plugin>
23 <groupId>org.apache.maven.plugins</groupId>
24 <artifactId>maven-compiler-plugin</artifactId>
25 <version>2.5.1</version>
26 <configuration>
27 <source>${java.version}</source>
28 <target>${java.version}</target>
29 </configuration>
30 </plugin>
31 <plugin>
32 <groupId>org.apache.maven.plugins</groupId>
33 <artifactId>maven-surefire-plugin</artifactId>
34 <version>2.4.2</version>
35 <configuration>
36 <skipTests>true</skipTests>
37 </configuration>
38 </plugin>
39 <plugin>
40 <groupId>org.apache.maven.plugins</groupId>
41 <artifactId>maven-war-plugin</artifactId>
42 <version>2.3</version>
43 <configuration>
44 <dependentWarExcludes>WEB-INF/lib/netty-*.jar,WEB-INF/lib/hibernate*.j
45 <archive>
46 <manifestEntries>

```

```

47         <dependencies>org.jboss.netty</dependencies>
48     </manifestEntries>
49 </archive>
50 </configuration>
51 </plugin>
52 </plugins>
53 </build>
54
55 <dependencies>
56 <dependency>
57 <groupId>org.jbpm</groupId>
58 <artifactId>jbpm-human-task-war</artifactId>
59 <version>5.4.0.Final</version>
60 <type>war</type>
61 </dependency>
62 </dependencies>
63
64 </project>

```

Tiene las mismas características que el pom.xml del console server, en relación a la exclusión de dependencias en el war y la generación del MANIFEST.MF.

Y, del mismo modo, debemos incluir un persistence.xml en un directorio src/main/resources/META-INF:

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <persistence version="2.0"
3   xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema
4   xsi:schemaLocation="
5     http://java.sun.com/xml/ns/persistence
6     http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
7   <persistence-unit name="org.jbpm.task" transaction-type="JTA">
8     <provider>org.hibernate.ejb.HibernatePersistence</provider>
9     <jta-data-source>java:jboss/datasources/jbpm5DS</jta-data-source>
10    <mapping-file>META-INF/Taskorm.xml</mapping-file>
11
12
13    <class>org.jbpm.task.Attachment</class>
14    <class>org.jbpm.task.BooleanExpression</class>
15    <class>org.jbpm.task.Comment</class>
16    <class>org.jbpm.task.Content</class>
17    <class>org.jbpm.task.Deadline</class>
18    <class>org.jbpm.task.Delegation</class>
19    <class>org.jbpm.task.EmailNotification</class>
20    <class>org.jbpm.task.EmailNotificationHeader</class>
21    <class>org.jbpm.task.Escalation</class>
22    <class>org.jbpm.task.Group</class>
23    <class>org.jbpm.task.I18NText</class>
24    <class>org.jbpm.task.Notification</class>
25    <class>org.jbpm.task.OnAllSubTasksEndParentEndStrategy</class>
26    <class>org.jbpm.task.OnParentAbortAllSubTasksEndStrategy</class>
27    <class>org.jbpm.task.PeopleAssignments</class>
28    <class>org.jbpm.task.Reassignment</class>
29    <class>org.jbpm.task.Status</class>
30    <class>org.jbpm.task.SubTasksStrategy</class>
31    <class>org.jbpm.task.Task</class>
32    <class>org.jbpm.task.TaskData</class>
33    <class>org.jbpm.task.User</class>
34    <properties>
35      <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect"/>
36      <property name="hibernate.max_fetch_depth" value="3"/>
37      <property name="hibernate.hbm2ddl.auto" value="update" />
38      <property name="hibernate.show_sql" value="true" />
39      <property name="hibernate.transaction.manager_lookup_class" value="org.hibernate.t
40
41    <!-- BZ 841786: AS7/EAP 6/Hib 4 uses new (sequence) generators which seem to cause
42    <property name="hibernate.id.new_generator_mappings" value="false" />
43
44    </properties>
45  </persistence-unit>
46 </persistence>

```

3.3. tnt-jbpm-gwt-console.

Es el más simple de todos puesto que no requiere más configuración que la del pom.xml:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMI
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/m
4   <modelVersion>4.0.0</modelVersion>
5
6   <parent>
7     <groupId>com.autentia.bpm.repository</groupId>
8     <artifactId>tnt-jbpm</artifactId>
9     <version>0.0.1-SNAPSHOT</version>
10  </parent>
11
12  <groupId>com.autentia.bpm.repository</groupId>
13  <artifactId>tnt-jbpm-gwt-console</artifactId>
14  <version>0.0.1-SNAPSHOT</version>
15  <packaging>war</packaging>
16
17  <inceptionYear>2013</inceptionYear>
18
19  <build>
20    <finalName>jbpm-console</finalName>
21    <plugins>
22      <plugin>
23        <groupId>org.apache.maven.plugins</groupId>
24        <artifactId>maven-compiler-plugin</artifactId>
25        <version>2.5.1</version>

```

```

26         <configuration>
27             <source>${java.version}</source>
28             <target>${java.version}</target>
29         </configuration>
30     </plugin>
31     <plugin>
32         <groupId>org.apache.maven.plugins</groupId>
33         <artifactId>maven-surefire-plugin</artifactId>
34         <version>2.4.2</version>
35         <configuration>
36             <skipTests>true</skipTests>
37         </configuration>
38     </plugin>
39     <plugin>
40         <groupId>org.apache.maven.plugins</groupId>
41         <artifactId>maven-war-plugin</artifactId>
42         <version>2.3</version>
43     </plugin>
44 </plugins>
45 </build>
46
47 <dependencies>
48     <dependency>
49         <groupId>org.jbpm</groupId>
50         <artifactId>jbpm-gwt-console</artifactId>
51         <version>5.4.0.Final</version>
52         <type>war</type>
53     </dependency>
54 </dependencies>
55 </project>

```

Lo único que hacemos es un overlay del war, no requiere persistencia porque solo hace uso de los servicios REST y la autenticación la configuramos en el siguiente punto.

3.4. Configuración de Jboss AS.

Ya vimos cómo configurar la [persistencia a MySQL](#) en un tutorial anterior y también necesitaríamos añadir la siguiente fuente de datos al fichero de configuración standalone.xml de Jboss

```

1  <datasource jta="true" jndi-name="java:jboss/datasources/jbpm5DS" pool-name="jbpmDS" e
2  <connection-url>jdbc:mysql://localhost:3306/jbpm5</connection-url>
3  <driver>mysql</driver>
4  <pool>
5      <min-pool-size>1</min-pool-size>
6      <max-pool-size>4</max-pool-size>
7      <prefill>>false</prefill>
8      <use-strict-min>>false</use-strict-min>
9      <flush-strategy>FailingConnectionOnly</flush-strategy>
10 </pool>
11 <security>
12     <user-name>jbpm5</user-name>
13     <password>jbpm5</password>
14 </security>
15 <validation>
16     <check-valid-connection-sql>SELECT 1</check-valid-connection-sql>
17     <validate-on-match>>false</validate-on-match>
18     <background-validation>>false</background-validation>
19 </validation>
20 </datasource>
21 <drivers>
22     <driver name="mysql" module="com.mysql">
23         <xa-datasource-class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-datasov
24     </driver>
25 </drivers>

```

Además de lo anterior debemos añadir una gestión de seguridad para autenticarnos en las aplicaciones web, para ello debemos añadir al mismo standalone.xml la siguiente configuración:

```

1  <security-domain name="jbpm-console" cache-type="default">
2      <authentication>
3          <login-module code="UsersRoles" flag="required">
4              <module-option name="usersProperties" value="${jboss.server.config.dir}/users
5              <module-option name="rolesProperties" value="${jboss.server.config.dir}/roles
6          </login-module>
7      </authentication>
8  </security-domain>

```

E incluir los ficheros users.properties en el mismo directorio de configuration, con el siguiente contenido:

```
1 admin=admin
```

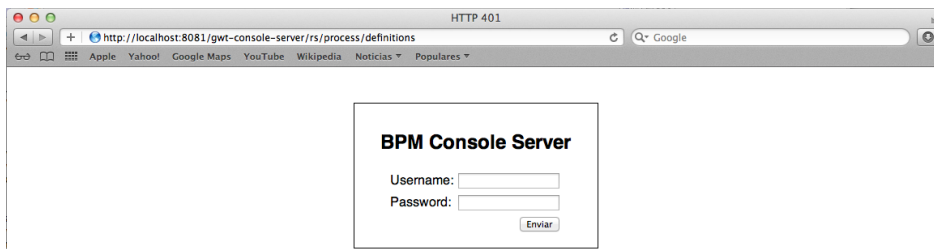
Y un fichero roles.properties con el siguiente contenido

```
1 admin=admin,manager,user
```

En este mismo standalone.xml podemos modificar el puerto al 8081:

```
1 <socket-binding name="http" port="8081"/>
```

Si todo va bien, accediendo a la siguiente dirección <http://localhost:8081/gwt-console-server/rs/process/definitions>, veremos una interfaz de autenticación básica



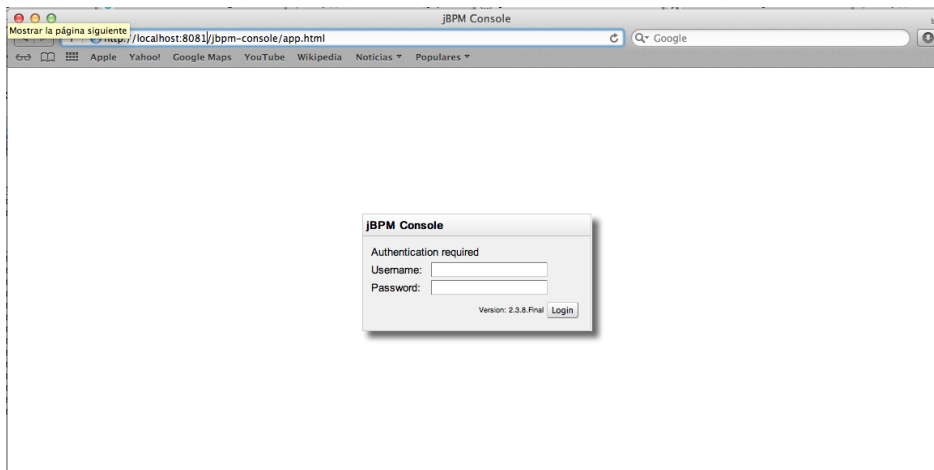
y tras incluir usuario y contraseña admin|admin veremos el resultado de la invocación al API

Accediendo a <http://localhost:8081/gwt-console-server/rs/server/resources/jbpm> podemos ver un listado de los servicios disponibles:

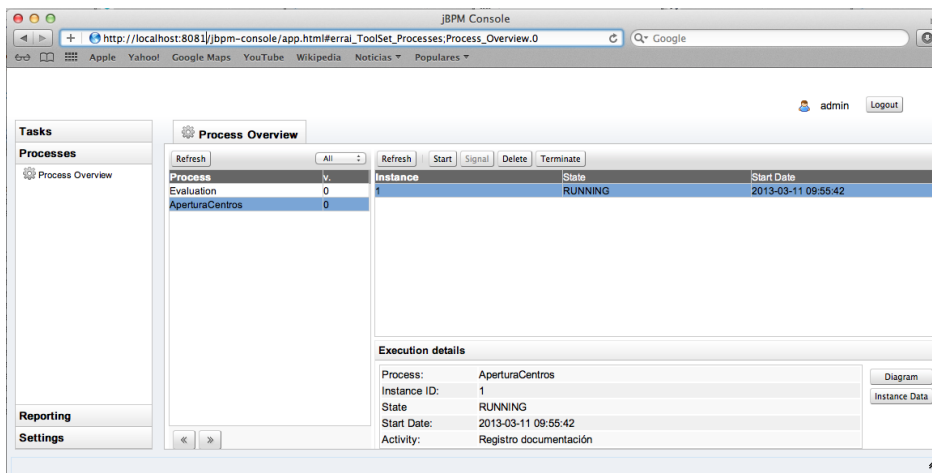
A screenshot of a web browser window showing a REST API services list. The browser address bar shows 'http://localhost:8080/gwt-console-server/rs/server/resources/jbpm'. The table lists various services with columns for Method, Path, Description, Consumes, and Produces.

Method	Path	Description	Consumes	Produce
Server Info				
<i>General REST server information</i>				
GET	/gwt-console-server/rs/server/status	*/	*/	application/json
GET	/gwt-console-server/rs/server/resources/{project}	*/	*/	text/html
Process Management				
<i>Process related data.</i>				
POST	/gwt-console-server/rs/process/definition/{id}/new_instance	*/	*/	application/json
POST	/gwt-console-server/rs/process/instance/{id}/state/{next}	*/	*/	application/json
GET	/gwt-console-server/rs/process/definitions	*/	*/	application/json
POST	/gwt-console-server/rs/process/definition/{id}/remove	*/	*/	application/json
GET	/gwt-console-server/rs/process/definition/{id}/instances	*/	*/	application/json
GET	/gwt-console-server/rs/process/instance/{id}/dataset	*/	*/	text/xml
POST	/gwt-console-server/rs/process/instance/{id}/end/{result}	*/	*/	application/json
POST	/gwt-console-server/rs/process/instance/{id}/delete	*/	*/	application/json
POST	/gwt-console-server/rs/process/tokens/{id}/transition	*/	*/	application/json
POST	/gwt-console-server/rs/process/tokens/{id}/transition/default	*/	*/	application/json
GET	/gwt-console-server/rs/process/definition/{id}/image	*/	*/	image/*
GET	/gwt-console-server/rs/process/definition/{id}/image/{instance}	*/	*/	image/*
GET	/gwt-console-server/rs/process/instance/{id}/activeNodeInfo	*/	*/	application/json
GET	/gwt-console-server/rs/process/definition/history/{id}/nodeInfo	*/	*/	application/json

Y accediendo a la siguiente dirección <http://localhost:8081/jbpm-console> veremos una interfaz de autenticación:



Y tras incluir usuario y contraseña admin|admin podremos acceder a la consola de procesos y bandeja de tareas:



5. Integración con LDAP.

Si una tarea está asignada a un grupo de usuarios, el jBPM Console Server se pondrá en contacto con el Human Task Server para solicitar un listado de los usuarios asociados a un grupo y comprobar su existencia.

Del mismo modo, la consola jBPM, para mostrar las potenciales tareas de un usuario, lleva a cabo una comprobación de los roles del usuario.

Para añadir usuarios y grupos a la autenticación de la consola jBPM no tenemos más que jugar con los ficheros `users.properties`:

```
1 rcanales=rcanales
2 jmsanchez=jmsanchez
```

y `roles.properties`:

```
1 rcanales=admin,manager,user,administrativos
2 jmsanchez=admin,manager,user,tramitadores
```

Y para configurar el servidor de tareas humanas para que su repositorio de usuarios sea nuestro servidor LDAP corporativo, tenemos que añadir un fichero `jbpm.usergroup.callback.properties` al directorio `src/main/resources` con un contenido similar al siguiente:

```
1 java.naming.provider.url=ldap://localhost:9898
2 ldap.bind.user=uid=admin,ou=system
3 ldap.bind.pwd=secret
4 ldap.user.ctx=ou=users,o=autentia
5 ldap.role.ctx=ou=groups,o=autentia
6 ldap.user.roles.ctx=ou=groups,o=autentia
7 ldap.user.filter=(uid={0})
8 ldap.role.filter=(cn={0})
9 ldap.user.roles.filter=(uniqueMember={0})
```

Apuntando a nuestro servidor ldap con autenticación si es necesario y los parámetros necesarios para que se puedan realizar las consultas de usuario y grupo.

Los parámetros que he expuesto son los necesarios para autenticarnos contra la estructura organizativa (ldif) que ya vimos en el tutorial anterior sobre [cómo hacer uso de un servidor LDAP embebido con el soporte de Spring Security](#).

Además de lo anterior deberíamos añadir un fichero `web.xml` al directorio `src/main/webapp/WEB-INF` con el contenido del original, pero modificando la entrada relativa a la clase que resuelve los grupos de un usuario

```
1 <init-param>
2   <param-name>user.group.callback.class</param-name>
3   <param-value>org.jbpm.task.identity.LDAPUserGroupCallbackImpl</param-value>
4 </init-param>
```

También debemos asegurarnos que haya una estrategia definida para las comunicaciones:

```
1 <init-param>
2   <param-name>active.config</param-name>
3   <param-value>hornetq</param-value>
4 </init-param>
```

Con todo ello y, según la configuración definida tanto a nivel de consola como a nivel de ldap, podríamos crear tareas para los grupos de usuarios tramitadores o administrativos.

Un último apunte: para que desde la consola de jBPM se pueda arrancar un nuevo proceso debemos disponer de un usuario **Administrator** en nuestro directorio activo.

```
1 dn: cn=admin,ou=groups,o=autentia
2 objectClass: groupOfUniqueNames
3 objectClass: top
4 objectClass: group
5 cn: admin
6 uniqueMember: cn=Administrator,ou=users,o=autentia
7
8 dn: cn=Administrator,ou=users,o=autentia
9 objectClass: organizationalPerson
10 objectClass: person
11 objectClass: inetOrgPerson
12 objectClass: top
13 cn: Administrator
14 sn: Administrator
```

```
15 uid: Administrator
16 mail: info@autentia.com
17 userPassword:: cGFzcw==
```

5. Referencias.

- <https://community.jboss.org/wiki/JPBM-530FinalManualDeploymentGuideForBeginner>
- <http://www.jboss.org/jbpm/roadmap>

6. Conclusiones.

Seguiremos jugando, solo nos queda la integración del generador de formularios dentro del ecosistema.

Si queréis jugar con nosotros [aquí](#) os dejo los proyectos mavenizados que se acumulan a los que teníamos para guvnor.

Un saludo.

Jose

jmsanchez@autentia.com

A continuación puedes evaluarlo:

[Regístrate para evaluarlo](#)



Por favor, vota +1 o compártelo si te pareció interesante



Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

» **Regístrate** y accede a esta y otras ventajas «



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Copyright 2003-2013 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) | [Contacto](#)

